
Manual de Instrucciones
Dream COMM
Dream COMM DC-320
UNIDAD DE DISCO
DC

INDICE

1.	Introducción	1
2.	Especificaciones	2
3.	Instalación	4
	Conexiones de los cables	4
	Encendido	4
	Inserción del diskette	4
4.	Uso de programas	5
	Uso de programas preelaborados	5
	El directorio del diskette	6
	Comparación de configuraciones y wild cards	7
5.	Comandos del diskette	9
	Lectura del canal de error	11
6.	Archivos secuenciales	13
7.	Archivos aleatorios	16
8.	Archivos relativos	23
	Uso de los archivos relativos	24
9.	Programación del controlador de diskette	27
10.	Cambio de número de dispositivo	30
	Método por software	30
	Método por hardware	
Apéndices		
A.	Lista de comandos	31
B.	Descripción de los mensajes de error	32

1) INTRODUCCION

La unidad de disco Drean Comm 320 es una unidad versátil y eficaz, construída para la serie Commodore de computadoras personales. Esta unidad es totalmente compatible con la computadora Drean Commodore 64 y reemplaza directamente la unidad de discos Commodore 1541, brindando un mejor rendimiento en términos de carga de datos, velocidad de grabación y capacidad de memoria intermedia (buffer).

Si usted es un principiante, los primeros capítulos le ayudarán a instalar y operar la unidad de discos. A medida que gana experiencia y habilidad, descubrirá otros usos de la unidad y en ese caso, los últimos capítulos serán de gran utilidad.

Si usted es un profesional con experiencia, este manual puede brindarle la información necesaria para aprovechar al máximo todas las características y potencia de la unidad de Disco Drean Comm 320.

Sea cual fuere su nivel de experiencia en programación, la unidad de Disco Drean Comm 320 aumentará considerablemente la eficiencia y capacidad de su sistema de computación.

Tenga en cuenta que este manual es una guía de referencia para la operación del Drean Comm 320. A pesar de que contiene instrucciones detalladas y una sección que le permite usar fácilmente paquetes de sofware preelaborados, deberá familiarizarse con el BASIC y los comandos que le ayudarán a operar la computadora y sus periféricos.

Recuerde que no es necesario que aprenda todos los datos contenidos en este manual al mismo tiempo. Los primeros tres o cuatro capítulos le permitirán usar la unidad de disco para la mayoría de las aplicaciones y los capítulos siguientes le indicarán cómo preparar un archivo, acceder a cualquier tipo de datos y programar la unidad de discos a nivel del lenguaje de la máquina.

NOTAS: En los ejemplos de FORMATO, las palabras en minúscula deben reemplazarse por una palabra o número adecuado que usted elija.

2) ESPECIFICACIONES

* Construcción de líneas esbeltas (de bajo perfil) y totalmente compatible con la Drean Commodore.
* Tamaño del diskette: 5 1/4 pulgadas de diámetro.
* Capacidad

Por diskette	174,8 kbytes
Entradas de directorio	144/diskette
Sector/Pista	17-21
Bytes/Sector	256
Pistas	35

* Indice MTBF promedio de 8000 horas.
* Requisitos de energía

Tensión	220/240 VCA
Frecuencia	50/60 Hertz
Disipación de energía	24 Watts

* Dimensiones mecánicas

Alto, largo, ancho	268 x 150 x 47,5 mm
Peso	2,8 kg.

UNIDAD DE DISCOS FLEXIBLES DREAN COMM 320

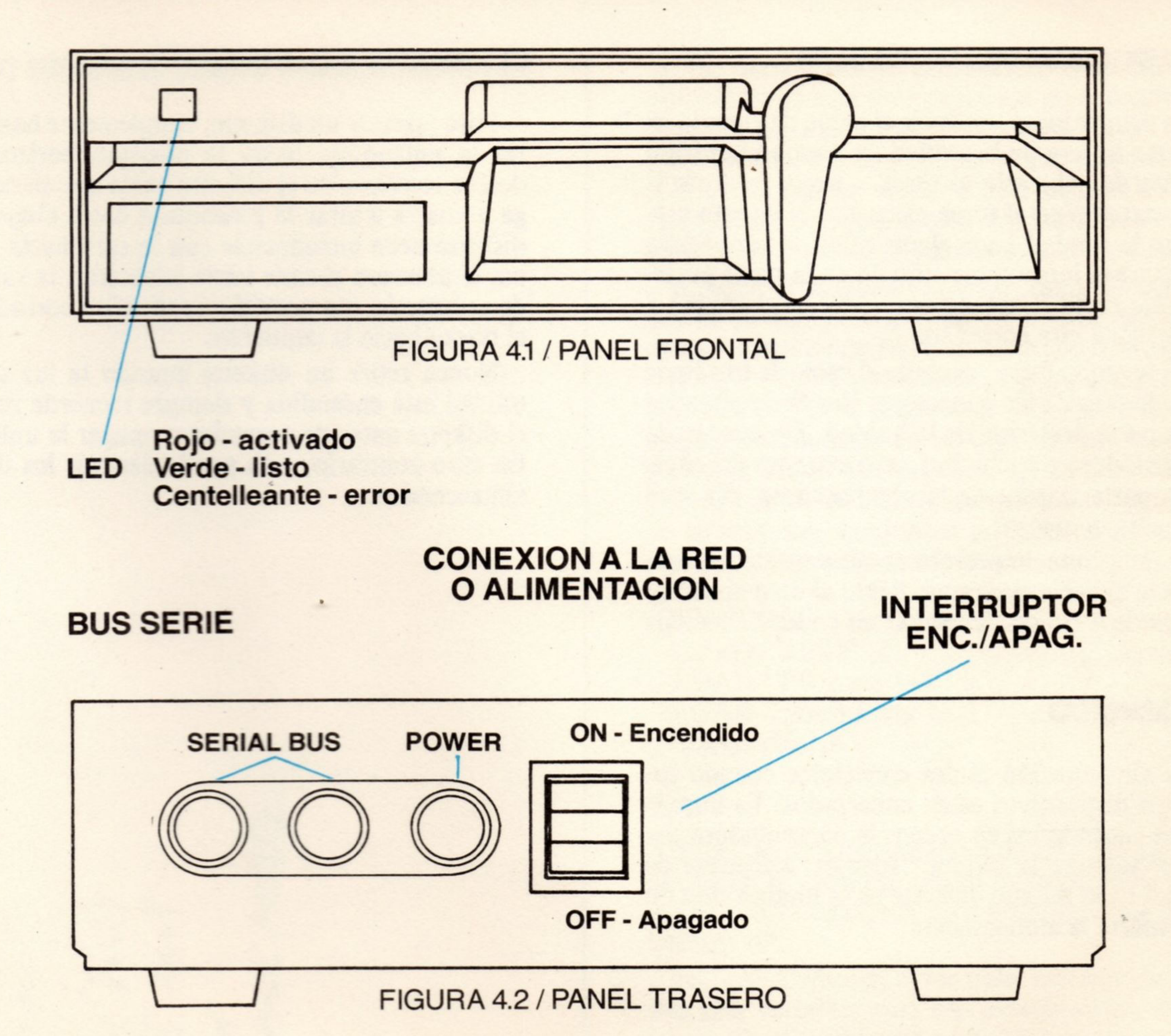


FIGURA 4.2 / PANEL TRASERO

Se ruega no efectuar ninguna conexión hasta que se haya completado la siguiente sección, de lo contrario ocasionará peligros o problemas al sistema.

3] INSTALACION

CONEXIONES DE LOS CABLES

En primer lugar, enchufe el cable de energía en la parte trasera de la unidad de disco; no entrará si trata de colocarlo al revés. Luego, enchufe el otro extremo en el toma eléctrico. Si en este momento la unidad hace algún ruido, desconéctelo mediante el interruptor situado en la parte posterior. No enchufe ningún otro cable en la unidad si la corriente está conectada.

En segundo lugar, enchufe el cable de Bus Serie en cada uno de los conectores Bus Serie situados en la parte posterior de la unidad. Desconecte la computadora y enchufe el otro extremo del cable en la parte trasera de la computadora. Ya está listo para comenzar.

Si tiene una impresora u otra unidad disco, conecte el cable correspondiente al otro conector Bus Serie a fin de "conectar en cadena" los dispositivos.

ENCENDIDO

La alimentación podrá conectarse cuando todos los dispositivos estén conectados. Es importantes encenderlos en orden: la computadora debe ser siempre la última. Además: asegúrese de que no haya ningún diskette en la unidad cuando desconecte la alimentación.

INSERCION DEL DISKETTE

Para insertar un diskette, simplemente haga girar la palanquita hasta la posición horizontal, deslice suavemente el diskette hasta que se detenga y vuelva a girar la palanquita hacia abajo. El diskette debe introducirse con la cara hacia arriba, la abertura grande hacia adelante y la ranura de protección (un pequeño cuadradito cortado en el disco) hacia la izquierda.

Nunca retire un diskette cuando la luz de la unidad esté encendida y siempre recuerde retirar el diskette antes de encender o apagar la unidad. En caso contrario, esta puede destruir los datos almacenados.

4) USO DE PROGRAMAS

USO DE PROGRAMAS PREELABORADOS

Si usted desea usar un programa ya grabado en un diskette, como por ejemplo un juego de video, a continuación le indicamos lo que debe hacer:

Haga girar la palanquita hacia arriba e inserte el diskette preprogramado, de manera que le etiqueta del diskette se encuentre hacia arriba y cerca de usted. Debe de haber una pequeña ranura en el diskette (posiblemente cubierta con una cinta adhesiva) que debe encontrarse a la izquierda. Haga girar la palanquita hacia abajo. Ahora, tipee el LOAD y el "nombre del programa" a cargar, luego pulse la tecla RETURN.

El diskette producirá un ruido y la pantalla indicará:

```
SEARCHING FOR NOMBRE DEL
PROGRAMA
LOADING
READY
```

Cuando la pantalla indica READY, tipee RUN y pulse la tecla RETURN; su programa está listo para usarse.

LOAD (COMANDO DE CARGA)

PROPOSITO: Para transferir un programa desde el diskette a la memoria de la computadora.

FORMATO: LOAD"nombre del programa", dispositivo, comando #

El nombre del programa es una serie de caracteres, es decir, ya sea un "nombre" o el contenido de una serie dada ya disponible. El número de dispositivo ya está preestablecido en el cuadro de circuitos del unidad de discos como 8. Si usted tiene más de una unidad, lea el capítulo sobre la forma de cambiar el número de dispositivo. En este manual se supone que usted utiliza 8 como número de dispositivo para la unidad de discos.

El número de comando es opcional. En caso de que no se indique o que sea cero, el programa se carga normalmente en el comienzo de la memoria disponible en su computadora para los programas BASIC. Si el número es 1, el programa se cargará exactamente en las mismas posiciones de memoria de las que provino. El número de comando 1 se utiliza principalmente para el lenguaje de la máquina, conjuntos de caracteres y otras funciones dependientes de la memoria.

EJEMPLOS:

```
LOAD "TEST",8
LOAD "Program = 1",8
LOAD "Mach Lang",8,1
LOAD A$,J,K
```

PRECAUCION: Además de colocar su programa en la memoria actual de la computadora, LOAD borra cualquier otro programa que se encuentre allí.

NOTA: Como en el ejemplo anterior, se pueden usar variables para representar series, números de dispositivos y números de comandos; simplemente asegúrese de que estén todos previamente definidos en su programa. Además, vea la nota sobre nombres de archivos en la página 9.

EL DIRECTORIO DEL DISKETTE

La unidad de discos es un dispositivo de acceso al azar (random access). Esto significa que el cabezal de lectura/grabación del dirve puede dirigirse a cualquier parte del diskette y acceder a cualquier bloque de datos de hasta 256 bytes de información. Hay 683 bloques en un dikette.

Afortunadamente, usted no tiene que preocuparse por los bloques individuales de datos (o de lo contrario puede consultar el capítulo 5). La unidad de discos cuenta con un programa llamado el Disk Operating System o DOS (Sistema Operativo del Dikette), que detecta los bloques por usted; los organiza en un Block Availability Map o BAM (Mapa de Disponibilidad de Bloques) y en un directorio. El BAM es simplemente una lista de verificación de bloques que se actualiza cada vez que se GUARDA (SAVE) un archivo ABIERTO (OPENED).

El directorio puede CARGARSE (LOAD) en la memoria de la computadora como un programa BASIC. Coloque el diskette en el drive y escriba:

LOAD“$”,8

La computadora indicará:

SEARCHING FOR $
LOADING
READY

Ahora, el directorio se encuentra en la memoria normal y si usted tipea LIST, se indicará en la pantalla. Para examinar el directorio desde un programa BASIC, vea el capítulo 6 relacionado con la instrucción GET #.

COMPARACION DE CONFIGURACIONES Y WILD CARDS

Para facilitar la CARGA (LOADING), la comparación de configuraciones le permite especificar ciertas letras en el nombre del programa, de manera que el primer programa del diskette que se corresponde con su configuración es el que se carga.

EJEMPLOS:
LOAD "*",8 (CARGA primero el archivo en el diskette)
LOAD "TE*",8 (CARGA primero el archivo que comienza con TE)
LOAD "TE??",8 (CARGA primero el archivo que tiene cuatro letras y que comienza con TE)
LOAD "T?NT",8 (CARGA primero el archivo que tiene cuatro letras pero puede ser TINT, TENT, etc.)

El asterisco (*) le indica a la computadora que no se preocupe por el resto del nombre, mientras que el singo de pregunta (??) actúa como un "wild card".

Lo antedicho también puede usarse al CARGAR (LOADING) el directorio en la memoria normal. Esto permite la búsqueda de una lista de programas específicos. El procedimiento es igual que el antedicho, excepto por la adición de "$:":

EJEMPLO:
LOAD "$:T?ST*",8 (CARGA todos los nombres de archivo del directorio que tengan la primera, tercera y cuarta letras correctas)

GUARDAR (SAVE)

PROPOSITO: Transferir un programa de la memoria normal al diskette para su uso posterior.
FORMATO: SAVE"nombre del programa", dispositivo #, comando #

Como se indicó anteriormente, el número del comando es opcional. Si ya hay en el diskette un programa o archivo con el mismo nombre, o si no hubiera suficiente espacio, se deberán borrar otros programas o se deberá usar un diskette diferente.

EJEMPLO:
SAVE"HOMEWORK",8

SAVE AND REPLACE (GUARDAR Y REEMPLAZAR)

PROPOSITO: Reemplazar un fichero ya existente por una versión revisada.
FORMATO: SAVE"@Ø:HOMEWORK",8

Si usted edita un programa ya existente y quiere guardarlo bajo el mismo nombre, la operación SAVE AND REPLACE lo hace automáticamente. Si usted desea guardar la versión antigua, guarde la nueva versión bajo un nombre diferente.

EJEMPLO:
SAVE"@Ø:HOMEWORK",8

VERIFY (VERIFICAR)

PROPOSITO Verifica el programa corriente con el que se encuentra en el diskette.
FORMATO: VERIFY"nombre del programa", dispositivo #, comando #

VERIFY realiza una comparación byte por byte del programa que se encuentra en la memoria corriente con la del diskette, como se especifica en el comando VERIFY.

EJEMPLO:
VERIFY"OLD VERSION",8

NOTA SOBRE LOS NOMBRES DE ARCHIVOS: Los nombres de los archivos deben comenzar con una letra y no con un número. Los espacios están permitidos. A pesar de que no hay ninguna restricción sobre la longitud del nombre de un archivo, todos los comandos deben tener una longitud de 58 caracteres o menos. Por ejemplo, en el comando VERIFY antedicho, hay 10 caracteres además del nombre del programa, de manera que en este caso, la longitud máxima del nombre es de 48 caracteres.

Además en el directorio, el largo máximo con el que figura el nombre del archivo es de 16 caracteres.

5) COMANDOS DE DISKETTE

Hasta ahora usted ha aprendido la forma sencilla de usar la unidad de discos. A fin de comunicarse con el diskette en forma más completa, deberá usar comandos de la unidad. Dos de ellos OPEN y PRINT # (ABRIR e IMPRIMIR), permiten la creación y llenado de un archivo de datos en un diskette. igual importancia tiene su capacidad de abrir un canal de comandos, permitiendo el intercambio de información entre la computadora y la unidad de discos.

OPEN (ABRIR)

PROPOSITO: Crea un archivo ABRIENDO (OPENING) un canal de comunicación entre la computadora y la unidad de disco.

FORMATO: OPEN archivo #, dispositivo #, (comando) canal #, serie de texto

El número de archivo debe ser cualquier número del 1 al 127. Los números del 128 al 255 pueden usarse aunque deben evitarse, debido a que hacen que la instrucción PRINT genere un avance de línea (line feed) después de un retorno de carro (carriage return). El número de dispositivo es generalmente 8.

El número de canal puede ser cualquier número del 2 al 15, que se refieren a los canales usados para comunicarse con el diskette y los canales 0 y 1 se utilizan por el sistema operativo para CARGAR (LOADING) y GUARDAR (SAVING). Los canales 2 al 14 pueden usarse para enviar a los archivos, mientras que el 15 se reserva como el canal de comandos.

La serie de texto es una serie de caracteres que se utiliza como nombre del archivo creado. Un archivo no puede crearse a menos que el nombre del archivo se especifique en la serie de texto. Si usted intenta abrir un archivo ya abierto, aparecerá la señal de error "FILE OPEN ERROR".

EJEMPLO:

OPEN 5,8,5 "TEST" (crea un archivo llamado TEST)

OPEN 15,8,15,"I" (envía un comando al diskette en el canal de comando)

OPEN A,B,C,Z$ (estas variables deben definirse)

PRINT = (IMPRESION)

PROPOSITO: Llena con datos un archivo previamente ABIERTO (OPENED)

FORMATO: PRINT # archivo #, serie de texto

El comando PRINT # funciona exactamente como el comando PRINT, excepto que los datos van a un dispositivo distinto que la pantalla, la unidad de discos en este caso. Cuando se lo utiliza con un canal de datos, PRINT # envía información a una memoria intermedia (buffer) en la unidad de discos que luego la CARGA (LOAD) en el diskette. Cuando se utiliza con un canal de comando, PRINT # envía comandos a la unidad de discos. El comando se coloca entre comillas como una serie de texto.

EJEMPLO:

PRINT # 7,C$ (llena el archivo 7 con la serie de texto C$)

PRINT # 15,"I" (envía el comando del diskette en el canal de comando)

INITIALIZE (INICIALIZAR)

PROPOSITO: Inicializa la unidad de discos a la condición de activación.
FORMATO: OPEN 15,8,15, "I" ó
OPEN 15,8,15 :
PRINT #15,"I"

A veces, una condición de error en el diskette le impedirá realizar una operación. INITIALIZE vuelve el drive de diskette a su estado original cuando se conecta la energía.

NEW (NUEVO)

PROPOSITO: Formatea el nuevo diskette o reformatea el usado.
FORMATO: PRINT #15, "NEW ∅: nombre del diskette, id #"

Este comando formatea un nuevo diskette. También es útil para borrar un diskette ya formateado, pues borra el diskette completo, coloca marcas de tiempo y bloques y crea el directorio y el BAM. El nombre del diskette sirve para la conveniencia del usuario, mientras que el id # es un identificador alfanumérico de 2 dígitos colocado en el directorio y en cada bloque del diskette. Si usted cambia diskettes mientras escribe datos, la unidad lo reconocerá mediante la verificación del id #.

EJEMPLO:
OPEN 15,8,15, "NEW ∅: TEST DISK, A1"
OPEN 15,8,15 : PRINT #15, "N ∅: MY DISK, MY"

Si el diskette debe borrarse pero no reformatearse, se utiliza el mismo comando pero deja de lado el id #.

EJEMPLO:
OPEN 15,8,15, "N ∅: NEW INFO"

SCRATCH (SUPRIMIR)

PROPOSITO: Borrar un archivo o archivos del diskette.
FORMATO: PRINT #15, "SCRATCH ∅: nombre del archivo"

Este comando borra uno o más archivos del diskette, habilitando espacio para archivos nuevos o más largos. Se pueden borrar grupos de archivos de una vez, nombrándolos todos juntos bajo un sólo comando de supresión.

EJEMPLO:
PRINT #15, "S∅: TEXT" (borra el archivo denominado TEXT)
PRINT #15,"SCRATCH∅: TEXT, ∅:TEST, ∅:MUSIC" (borra los archivos TEXT, TEST y MUSIC)

COPY (COPIAR)

PROPOSITO: Duplicar un archivo existente.
FORMATO: PRINT #15, "COPY ∅: nuevo nombre del archivo = ∅ : nombre antiguo del archivo"

COPY le permite hacer una copia de cualquier programa o archivo en el diskette. El nuevo nombre del archivo debe ser distinto del antiguo. COPY también puede combinar hasta cuatro archivos en uno nuevo.

EJEMPLO:
PRINT #15,"C Ø:BACKUP = Ø : ORIGINAL"
PRINT # 15,"COPY Ø :NEWFILE = Ø:OLD1, Ø: OLD2,Ø" (combina OLD1 y OLD2 en NEWFILE)

RENAME (CAMBIAR EL NOMBRE)

PROPOSITO: Cambiar el nombre de un archivo existente.
FORMATO: PRINT #15,"RENAMEO: nuevo nombre = Ø: nombre antiguo"

Este comando le permite cambiar el nombre de un archivo una vez que se encuentre en el directorio del diskette. RENAME no funcionará con ningún archivo que se encuentra actualmente abierto.
EJEMPLO:
PRINT 15,"R Ø:GOODNAME = Ø: DUMBNAME" (nombre correcto = Ø: nombre ficticio)

VALIDATE (VALIDAR)

PROPOSITO: Elimina los espacios desperdiciados del diskette.
FORMATO: OPEN 15,8,15,"VØ:"

Después de guardar y borrar varios archivos de un diskette, comienzan a acumularse en el mismo pequeños espacios en los datos y espacios desperdiciados de memoria. VALIDATE reorganiza el diskette a fin de que pueda aprovechar la memoria al máximo en el espacio disponible. También este comando elimina archivos ABIERTOS (OPENED) que nunca fueron CERRADOS (CLOSED) adecuadamente.

PRECAUCION: VALIDATE borra los archivos aleatorios (vea el capítulo 7). Si su diskette contiene archivos aleatorios, NO use este comando.

LECTURA DEL CANAL DE ERROR

Sin el DOS Support Program (Programa de Apoyo del DOS), no hay ninguna posibilidad de leer el canal de error del diskette, ya que se necesita leer el comando INPUT #, que no puede utilizarse fuera de un programa. A continuación se indica un programa BASIC para leer el canal de error:

```
10 OPEN 15,8,15
20 INPUT #15, A$, B$, C$, D$
30 PRINT A$, B$, C$, D$
```

Cuando usted utiliza un INPUT # (ENTRADA) del canal de comandos, lee hasta cuatro variables que describen la condición de error. La primera, tercera y cuarta son números, de manera que pueden usarse variables numéricas. Las entradas (inputs) están organizadas de la siguiente forma:

Primera: número de error (Ø significa ningún error).
Segunda: descripción del error.
Tercera: número de pista donde se produjo el error.
Cuarta: bloque (sector) en la pista donde se produjo el error.

Los errores en la pista 18 se refieren al BAM y al directorio.

CLOSE (CIERRE)

PROPOSITO: Asignación correcta de los bloques de datos, cierra la entrada.
FORMATO: CLOSE archivo #

Este comando es muy importante. Una vez que un archivo que se abrió no se necesita más para entrada de datos, DEBE CERRARSE O DE LO CONTRARIO TODOS LOS DATOS EN ESE ARCHIVO SE PERDERAN.

Es muy importante que los archivos de datos se CIERREN (CLOSE) antes de que se cierre el canal de error (canal # 15). De lo contrario la unidad de discos lo cerrará por usted pero el BASIC aún pensará que están abiertos y le permitirá grabar en ellos. El canal de error debe ABRIRSE (OPEN) primero y CERRARSE (CLOSE) después de todos sus archivos.

NOTA: Si su programa BASIC produce una condición de error, todos los archivos se CIERRAN en BASIC, pero no en la unidad de discos. Esto es MUY PELIGROSO. Inmediatamente tipee:

```
CLOSE 15: OPEN 15,8,15: CLOSE 15
```

Esto reinicializará su unidad y guardará todos los archivos.

6) ARCHIVOS SECUENCIALES

Los archivos secuenciales se almacenan y leen en forma secuencial desde el comienzo hasta el fin. Existen básicamente tres tipos diferentes de archivos secuenciales que pueden usarse. El primero es el archivo del programa que se abrevia en el directorio como PRG. El PRG es el único archivo secuencial que puede almacenar y leer programas. El segundo archivo, secuencial (SEQ) y el tercer archivo, usuario (USR), se utilizan para el manejo de datos. Estos dos archivos deben abrirse como el canal de comando descrito en el capítulo anterior.

OPEN (ABRIR)

PROPOSITO: Abrir un archivo secuencial.
FORMATO: OPEN archivo # , dispositivo # , canal # , "Ø: nombre, tipo, dirección"

El número de archivo es el mismo que en los usos anteriores del comando OPEN, el número de dispositivo es generalmente 8, el número de canal es un canal de datos, 2 a 14. Es apropiado usar el mismo número para el archivo y el canal, a fin de recordarlo con facilidad (posiblemente usted notó esta característica en los ejemplos anteriores).

El nombre corresponde al nombre de archivo, para el que no pueden usarse "wild cards" o comparación de configuraciones si se está creando un archivo de grabación. El tipo puede ser cualquiera de la lista indicada más abajo, o por lo menos la primera letra de uno de ellos. La dirección debe ser READ (LEER) o WRITE (GRABAR), o por lo menos las primeras letras de los términos.

TIPO DE ARCHIVO	SIGNIFICADO
PRG	Archivo de programa
SEQ	Archivo secuencial
USR	Archivo de usuario
REL	Relativo (no implementado en BASIC 2.0)

EJEMPLOS:
OPEN 5,8,5, "Ø: DATA, S, R"
OPEN A,B,C, "Ø: TEXT, P,W,"
OPEN A,B,C, "Ø:" + A$ + "U,W" (ABRE un archivo de grabación con un nombre especificado por la variable en serie A$)
OPEN 2,8,2 "@Ø: PHONES, S, W"
(reemplaza la versión antigua del archivo por una nueva)

Una vez que el archivo fue abierto para leer o grabar, pueden usarse tres comandos para transferir los datos. Estos comandos son PRINT#, INPUT#, y GET#.

PRINT # (IMPRIMIR)

PROPOSITO: Dirige la salida al archivo previamente abierto.
FORMATO: PRINT# archivo#, lista de datos (no se debe dejar espacio entre PRINT y #)

La instrucción PRINT# trabaja exactamente como PRINT: capacidades de formateo para los tipos de puntuación y datos funcionan de igual manera, pero esto también significa que tiene que ser cuidadoso al ingresar datos en los archivos. El número de archivo es el que recién se ABRIO (OPENED) y la lista de datos consiste en variables y/o texto dentro de las comillas.

Se debe tener cuidado al grabar datos, a fin de que resulte lo más fácil posible leerlos posteriormente. Las comas usadas para separar items harán que se almacenen espacios en el diskette. Los punto y coma evitarán que se almacenen espacios. Si las comas y los punto y coma están ausentes, se almacenará un retorno de carro (CR) al final de los datos escritos. Observe el siguiente programa modelo:

```
10 A$ = "THIS IS A"
20 B$ = "TEST"
30 OPEN 8,8,8, "Ø: TEST#, S,W"
40 PRINT#8,A$,B$"OF THE DISK"
50 CLOSE 8
60 END
```

Si usted pudiera observar los datos y su posición en el diskette, se leerían como sigue:

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
THIS IS A                    T E S T O F T H E   D I S K CReof (fin del archivo)
```

La coma, punto y coma y retorno del carro tienen un significado especial cuando se los almacena en el diskette. Cuando se usan dentro de una serie de comillas, se almacenan como caracteres comunes. Cuando se los usa como separadores entre campos, la coma inserta espacios (generalmente un desperdicio de memoria), el punto y coma no lo hace, y el CR almacena un retorno de carro en el diskette. Estas características son importantes cuando usted usa GET# o INPUT# para recuperar los datos almacenados.

GET # (OBTENER)

PROPOSITO: Obtener datos del diskette byte por byte.
FORMATO: GET # archivo #, lista variable.

Los datos se obtienen byte por byte, incluyendo CRs, comas, y otros separadores. Generalmente, es más seguro usar variables de series de caracteres a fin de evitar mensajes erróneos.

EJEMPLOS:

GET # 8, A$

GET # 5, A (solamente funciona para datos numéricos)

GET # A, B$, C$, D$ (OBTIENE más de una variable por vez)

La instrucción GET # es muy útil cuando se desconoce la estructura o contenido propiamente dichos de datos, como un archivo en un diskette dañado. Si usted está familiarizado con el archivo y no hay problemas, INPUT # es más eficaz. Pero para observar datos en un archivo dañado o no conocido, el siguiente programa modelo leerá el contenido (en este caso, del archivo creado en el programa modelo de PRINT #).

```
10 OPEN 8,8,8, "TEST"
20 GET # 8,A$:PRINT A$;
30 IF ST=Ø THEN 20 (ST es una señal de
   estado)
40 CLOSE 8
50 END
```

INPUT # (ENTRADA)

PROPOSITO: Recupera los datos del diskette en grupos.
FORMATO: INPUT # archivo #, variable

El número de archivo es igual al ABIERTO (OPENED) y la variable puede representar series de caracteres o números. Para leer un grupo de datos, se necesitan separadores para indicar el comienzo y finalización del grupo. Estos son la coma, punto y coma y CR, y funcionan en la forma indicada en la sección correspondiente al comando PRINT #. Los números se almacenan con un espacio frente a ellos que está vacío para números positivos y contiene un signo negativo para números negativos. A continuación se indica un programa modelo:

```
10 OPEN 8,8,8,"@Ø: DATAFILE,S,W"
20 FOR A=1 TO 10
30 PRINT # 8,A
40 NEXT A
50 CLOSE 8
60 OPEN 2,8,2,"DATAFILE"
70 INPUT # 2,B : PRINT B
80 IF ST=0 THEN 70
90 CLOSE 2
100 END
```

Este programa modelo grabará los números 1 hasta 10 en un archivo secuencial denominado DATAFILE. Las líneas 70 y 80 leerán e imprimirán los datos del diskette. Vea la página 21 donde encontrará dos programas modelo muy útiles.

7) ARCHIVOS ALEATORIOS

Los archivos secuenciales son convenientes cuando usted trabaja con un flujo continuo de datos, pero algunos trabajos requieren más flexibilidad. Por ejemplo, si usted tiene una gran lista de direcciones, es incómodo explorar la lista completa para encontrar la dirección de una persona. Un método de acceso al azar le permitirá seleccionar los datos deseados sin necesidad de leer el archivo completo.

Existen dos tipos de archivo que pueden realizar esto: archivos aleatorios y archivos relativos. Los archivos aleatorios son la mejor elección cuando la velocidad es un factor importante, como es el caso de programas de lenguaje de máquinas. Esto se debe a que el programa mantiene las ubicaciones de los datos cuando se emplean los archivos aleatorios, mientras que el DOS mantiene las ubicaciones de los archivos relativos. El problema es que los archivos aleatorios son fáciles de eliminar del diskette en forma accidental, ya que el DOS no los mantiene.

Los archivos aleatorios son archivos que han sido grabados en una ubicación física determinada en el diskette. El diskette se divide en 35 anillos concéntricos o pistas y cada una de estas pistas contiene entre 17 y 21 sectores.

NUMERO DE PISTA	GAMA DE SECTORES	TOTAL DE SECTORES
1 A 17	0 A 20	21
18 A 24	0 A 18	19
25 A 30	0 A 17	18
31 A 35	0 A 16	17

Es posible leer y grabar en cualquier bloque del diskette, así como determinar qué bloques están disponibles para usarse. Los siguientes comandos explican cómo usar las funciones de los archivos aleatorios.

OPEN (ABRIR)

PROPOSITO: ABRE un canal de datos para acceso al azar.
FORMATO: OPEN archivo # , dispositivo # , canal # , " # "

Al trabajar con archivos aleatorios, usted necesita tener dos canales abiertos al diskette: el canal de comando (15) para enviar comandos y el canal de datos (2 a 14) para la transferencia de datos. El canal de datos para los archivos de acceso al azar se ABRE seleccionando el signo " # ", como el nombre del archivo.

El " # " adicional al final del comando hace que el diskette asigne una memoria intermedia (buffer) de 256 bytes para manejar el bloque deseado de datos. Si se especifica un número de memoria intermedia (buffer), la memoria intermedia asignada será la que usted especificó.

EJEMPLOS:

OPEN 5,8,5," # " (a usted no le importa cuál memoria intermedia)

OPEN A,B,C," # 2" (usted especifica la memoria intermedia 2)

BLOCK-READ (LECTURA-BLOQUE)

PROPOSITO: Leer un bloque específico de datos del diskette.
FORMATO: PRINT # archivo # , "BLOCK-READ:" canal # , drive # , pista # , bloque # (BLOCK-READ puede ser reemplazado por B-R)

Los números de archivo y canal son los que se han ABIERTO (OPENED). El número de pista y número de bloque indican qué bloque de 256 bytes debe leerse. La ejecución de este comando hace que la unidad de discos mueva el bloque especificado de datos en el área de la memoria intermedia (buffer). Los datos pueden entonces leerse de esta área usando INPUT # o GET # . Solamente se leerán los datos en ese bloque en particular y todos los bytes no usados en el bloque no se podrán leer. El programa modelo que se indica a continuación utiliza BLOCK-READ para leer el contenido del bloque 9 en la pista 5 y mostrar en pantalla su contenido.

```
10 OPEN 15,8,15
20 OPEN 8,8,8," # "
30 PRINT # 15, "B-R:"8,0,5,9
   (lee el bloque en el buffer)
40 GET # 8, A$
50 PRINT A$
60 IF ST = 0 THEN 40
70 PRINT "READ COMPLETE"
80 CLOSE 8 : CLOSE 15
```

BLOCK-WRITE (GRABACION-BLOQUE)

PROPOSITO: Grabar un bloque de datos en una ubicación especificada de bloque en el diskette.
FORMATO: PRINT # archivo # , "BLOCK-WRITE:" drive # , canal # , pista # , bloque #

BLOCK-WRITE puede abreviarse B-W. Este comando hace que los datos previamente almacenados en la memoria intermedia (buffer) se escriban en la ubicación especificada en el diskette. Los datos deben transferirse a la memoria intermedia (buffer) en un canal de datos usando PRINT# antes de GRABARLO EN BLOQUE (BLOCK WRITING) en el diskette. El DOS verifica cuántos bytes se almacenan en la memoria intermedia y almacena el conteo de bytes en el primer byte del bloque cuando se ejecuta BLOCK-WRITE. Esto significa que solamente pueden leerse o grabarse 255 bytes en el bloque, ya que el conteo de bytes usa el primer byte del bloque. A continuación se indica un ejemplo de una operación de rutina que grabe datos al mismo bloque que se lee en el ejemplo de BLOCK-READ antedicho (pista 5, bloque 9):

```
10 OPEN 15,8,15
20 OPEN 8,8,8,"#"
30 FOR AA=1 A 32
40 PRINT#8, "TESTING"
50 NEXT
60 PRINT#15, "B-W:" 8,0,5,9
70 CLOSE 8 : CLOSE 15
```

BLOCK-ALLOCATE (ASIGNACION-BLOQUE)

PROPOSITO: Determinar si un bloque en particular está libre y en ese caso asignarlo.

FORMATO: PRINT#15,"B-A" canal#, drive#, pista#, bloque#

Según se mencionó anteriormente, el DOS no mantiene el diskette cuando se utiliza BLOCK-READ y BLOCK-WRITE. Pero el usuario puede asegurarse de que un bloque en particular está disponible, utilizando el comando BLOCK-ALLOCATE. Esto permite usar los comandos de BLOQUE (BLOCK) en un diskette que ya tiene archivos. Mediante la verificación del BAM, el comando determina si el bloque especificado fue usado. Debido a que el BAM se actualiza cada vez que un archivo se almacena en el diskette, los archivos pueden mantenerse. Los comandos de BLOQUE (BLOCK) no actualizan el BAM y en consecuencia no serán reconocidos, a menos que se haya ejecutado BLOCK-ALLOCATE (ASIGNACION-BLOQUE).

PRECAUCION: el comando VALIDATE no reconoce archivos aleatorios y nunca debe usarse en un diskette que contenga este tipo de archivos.

Si BLOCK-ALLOCATE (ASIGNACION-BLOQUE) determina que el bloque especificado ya se ha usado, se generará una señal de error (65). El mensaje de error le indica los números de la próxima pista y bloque disponibles en el diskette. Este bloque no se asigna, de manera que el comando BLOCK-ALLOCATE (ASIGNACION-BLOQUE) debe usarse nuevamente, pero

esta vez usted puede estar seguro de que el bloque especificado está libre para usarse. El siguiente programa asignará un bloque y grabará en el mismo. Si el bloque ya ha sido usado, grabará en el próximo disponible, como se indica en el mensaje de error.

```
10 OPEN 15,8,15:OPEN 8,8,8,"#"
20 PRINT#8, 'THIS GOES INTO THE
   BUFFER"
30 T=5 : S=9
40 PRINT#15,"B-A:"O,T,S
50 INPUT#15,A,A$,B,C
60 IF A=65 THEN T=B:S=C : GOTO 40
70 PRINT#15,"B-W:" 8,0,T,S
80 PRINT"DATA WAS STORES IN
   TRACK:"T," SECTOR:"S
90 CLOSE 8:CLOSE 15
100 END
```

La línea 20 carga la memoria intermedia (buffer) con el texto, las líneas 30 y 40 verifican el bloque 9 en la pista 5 para ver si está libre, y la línea 50 ingresa la señal de error. Si el bloque está libre, los datos se almacenan allí. Si ya se ha usado el bloque 9 en la pista 5, la línea 60 toma los números del nuevo bloque y pista y asigna el bloque que especifican; luego los datos se almacenan en el nuevo bloque. Las líneas 70 y 80 leen los números de pista y bloque en la computadora y los muestran en la pantalla.

BLOCK-FREE (LIBERACION-BLOQUE)

PROPOSITO: Liberar un bloque usado para un nuevo uso.
FORMATO: OPEN 15,8,15,"B-F:" drive, pista, bloque

Este comando es el opuesto a BLOCK-ALLOCATE (ASIGNACION-BLOQUE), debido a que libera un bloque que usted no quiere usar más para que sea usado por el sistema. Es similar al comando SCRATCH (SUPRIMIR) en el sentido que no borra nada sino que simplemente libera la entrada, en el BAM en este caso.

EJEMPLOS:

```
10 OPEN 8,8,"#"
20 OPEN 15,8,15,"B-F:"0,5,9
30 CLOSE 8 : CLOSE 15
```

(libera la pista 5, bloque 9 para usarse)

BUFFER-POINTER (INDICADOR-MEMORIA INTERMEDIA)

PROPOSITO: Permitir el acceso al azar dentro de un bloque.
FORMATO: PRINT#15,"B-P:" canal#, ubicación (byte#)

El indicador de la memoria intermedia detecta donde se grabó el último dato e indica donde se leerá el próximo segmento de datos. Al cambiar la ubicación del indicador de la memoria intermedia (buffer) en esta última, se puede acceder al azar a los bytes individuales dentro de un bloque. Esto significa que se puede dividir un bloque en registros.

EJEMPLO:

PRINT # 15,“B-P:” 5,64 (coloca el indicador en el caracter 64 de la memoria intermedia (buffer).

USO DE LOS ARCHIVOS ALEATORIOS

El problema con los archivos aleatorios es que no hay ninguna forma de verificar en qué bloque han sido usados. Para verificar esto, el método más común es crear un archivo secuencial para cada archivo aleatorio. Este archivo sirve para mantener una lista de las ubicaciones de registros, pistas y bloques. Esto significa que usted tiene tres canales abiertos al diskette para cada archivo aleatorio: El canal de comando, el canal para los datos aleatorios y el canal para el archivo secuencial. Además usted utiliza dos memorias intermedias (buffers) al mismo tiempo.

A continuación encontrará cuatro programas que utilizan acceso al azar dentro de bloques:

PROGRAMA A graba 10 bloques de acceso al azar con un archivo secuencial.

PROGRAMA B lee el mismo archivo.

PROGRAMA C graba 10 bloques de acceso al azar con 4 registros cada uno.

PROGRAMA D lee el mismo archivo.

PROGRAMA A: GRABA UN ARCHIVO SECUENCIAL

```
10 OPEN 15,8,15
20 OPEN 5,8,5,“   ”
30 OPEN 4,8,4,“@O:KEYS,S,W”
40 A$ = “Contenido del registro $”
50 FOR R = 1 A 10
60 PRINT # 5,A$“,”R
70 T = 1:S = 1
80 PRINT # 15,“B-A:”O,T,S
90 INPUT # 15,A,B$,C,D
100 IF A = 65 THEN T = C:S = D : GOTO 80
110 PRINT # 15,“B-W:”5,O,T,S
120 PRINT # 4,T“,”S
130 NEXT R
140 CLOSE 4:CLOSE 5:CLOSE 15
```

PROGRAMA B: LEE UN ARCHIVO SECUENCIAL

```
10 OPEN 15,8,15
20 OPEN 5,8,5,“ # ”
30 OPEN 4,8,4,“KEYS,S,R”
40 FOR R = 1 A 10
50 INPUT # 4,T,S
60 PRINT # 15,“B-R:”5,0,T,S
70 INPUT # 5,A$,X
80 IF A$ < > “Contenido de registro # ”
   O X < > R THEN STOP
90 PRINT # 15,“B-F:”O,T,S
100 NEXT R
110 CLOSE 4:CLOSE 5
```

```
120 PRINT#15,"SO:KEYS"
130 CLOSE 15
```

PROGRAMA C: GRABA ARCHIVOS DE ACCESO AL AZAR

```
10 OPEN 15,8,15
20 OPEN 5,8,5,"#"
30 OPEN 4,8,4,"KEYS,S,W"
40 A$="Contenido de Registro #"
50 FOR R=1 A 10
60 FOR L=1 A 4
70 PRINT#15,"B-P:"5: (L-1)*64
80 PRINT#5,A$",''L
90 NEXT L
100 T=1 : S=1
110 PRINT#15,"B-A:",O,T,S
120 INPUT#15,A,B$,C,D
130 IF A=65 THEN T=C:S=D:GOTO 110
140 PRINT#15,"B-W:"5,O,T,S
150 PRINT#4,T",''S
160 NEXT R
170 CLOSE 4:CLOSE 5:CLOSE 15
```

PROGRAMA D: LEE ARCHIVO ALEATORIO

```
10 OPEN 15,8,15
20 OPEN 5,8,5,"#"
30 OPEN 4,8,4,"KEYS,S,R"
40 FOR R=1 A 10
50 INPUT#4,T,S
60 PRINT#15,"B-R:"5,O,T,S
70 FOR L=1 TO 4
80 PRINT#15,"B-P:"5,(L-1)*64
90 INPUT#5,A$,X
100 IF A$<>"Contenido de registro #"
    OR X=L THEN STOP
110 NEXT L
120 PRINT#15,"B-F:"0,T,S
130 NEXT R
140 CLOSE 4:CLOSE 5
150 PRINT#15,"SO:KEYS"
160 CLOSE 15
```

USER1 (USUARIO 1)

PROPOSITO: Leer un bloque completo de 256 bytes del diskette a la memoria intermedia (buffer).

FORMATO: PRINT# archivo # , "U1:" canal # , drive # , pista # , bloque #

El comando USER1 es prácticamente idéntico al comando BLOCK-READ (LECTURA-BLOQUE) excepto que USER1 fuerza el indicador de la memoria intermedia (buffer) hacia el fin del bloque que debe leerse, a fin de que se lea todo el bloque. USER1 puede abreviarse como U1 ó UA. A continuación se indica un programa modelo que obtendrá los 256 bytes completos de la pista 5, bloque 9 y la mostrará en la pantalla.

```
10 OPEN 15,8,15: OPEN 8,8,8
20 PRINT# 15, "U1:"8,0,5,9
30 GET# ,A$:PRINT A$;
40 IF ST = 0 THEN 30
50 CLOSE 8: CLOSE 15
60 END
```

USER2 (USUARIO 2)

PROPOSITO: Grabar un bloque de datos en el diskette sin alterar el indicador-memoria intermedia (buffer)

FORMATO: PRINT# 15,"U2:" canal # , drive # , pista # , bloque #

USER2 (abreviado como U2 ó UB) es muy similar al comando BLOCK-WRITE (GRABACION-BLOQUE), pero U2 no cambia la posición del indicador-memoria intermedia (buffer-pointer) cuando la memoria intermedia (buffer) se graba en el diskette. Esto es muy útil si usted quiere leer un bloque de datos en la memoria intermedia (buffer) y modificarlo. Después de encontrar los datos pertinentes con el indicador-memoria intermedia (buffer-pointer) y modificarlos, el comando USER2 puede usarse para regrabar los datos en el diskette, y para esto, el indicador-memoria intermedia (buffer-pointer) estará en la posición correcta. Si se utilizó BLOCK-WRITE (GRABACION-BLOQUE), el indicador-memoria intermedia (buffer-pointer) tendrá que reajustarse primero. El siguiente programa utiliza los comandos USER1 y USER2.

```
10 OPEN 15,8,15: OPEN 8,8,8
20 PRINT # 15,"U1:"8,0,5,9
30 PRINT # 15,"B-P:"8,32
40 PRINT# 8, "A"
50 PRINT # 15,"U2:"8,0,5,9
60 CLOSE 8: CLOSE 15
70 END
```

La línea 20 lee pista 5 bloque 9 en la memoria intermedia (buffer).

La línea 30 mueve el indicador-memoria intermedia (buffer-pointer) al byte 32.

La Línea 40 cambia el byte 32 al carácter "A".

La Línea 50 graba la memoria intermedia (buffer) nuevamente en el diskette.

Aún cuando el indicador-memoria intermedia (buffer-pointer) haya sido alterado, USER2 se asegura de que el indicador-memoria intermedia (buffer-pointer) antiguo no se cambie en el diskette.

8] ARCHIVOS RELATIVOS

Los archivos relativos pueden tener acceso a cualquier segmento de datos en el diskette como los archivos aleatorios, pero usted no tiene que mantener los archivos en su propio programa. El DOS mantiene los datos para usted, verificando el estado de los archivos. Por este motivo, los archiso relativos son más lentos que los archivos aleatorios, pero con frecuencia una mayor conveniencia compensa esto.

El DOS verifica las pistas y sectores (bloques) usados, y hasta permite que los registros se superpongan de un bloque al otro. Esto se efectúa estableciendo sectores laterales, una serie de indicadores para el comienzo de cada registro. Puede haber 6 sectores laterales en un archivo, y cada sector lateral puede indicar hasta 120 registros. Esto significa que un archivo puede tener hasta 720 registros y puesto que cada registro puede tener una longitud de 254 caracteres, un archivo puede llenar todo el disco.

El formato del bloque consiste en los primeros dos bytes que especifican la pista y sector del próximo bloque de datos. Los próximos 254 bytes contienen los datos propiamente dichos. Cualquier registro vacío tendrá FF (hexadecimal para todos los 1) en el primer byte y 00 en el resto del registro. Los sectores laterales se utilizan para referenciar todas las ubicaciones de los sectores laterales, no solamente las 120, ubicaciones de bloques de datos relacionadas con dicho sector lateral.

FORMATO DE LOS ARCHIVOS RELATIVOS

BLOQUE DE DATOS:

BYTE	DEFINICION
0,1 ...	Pista y sector del próximo bloque de datos.
2-256...	254 bytes de datos. Los registros vacios contienen FF (todos binarios) en el primer byte seguido por 00 hasta el final del registro. Los registros parcialmente llenados se rellenan con ceros (00).

BLOQUE DEL SECTOR LATERAL:

BYTE	DEFINICION
0,1	Pista y sector del próximo bloque del sector lateral.
2	Número del sector lateral (0-5)
3	Longitud del registro.
4,5	Pista y sector del primer sector lateral (0).
6,7	Pista y sector del segundo sector lateral (1).
8,9	Pista y sector del tercer sector lateral (2).
10,11 ...	Pista y sector del cuarto sector lateral (3).
12,13 ...	Pista y sector del quinto sector lateral (4).
14,15	Pista y sector del sexto sector lateral (5).
16-256 .	Indicadores de pista y sector hasta 120 bloques de datos.

USO DE LOS ARCHIVOS RELATIVOS

Los archivos relativos se crean la primera vez que se abren (OPEN), y se usarán hasta que se cierren (CLOSED). Un archivo relativo solamente puede borrarse de un diskette utilizando el comando SCRATCH (SUPRESION), o reformateando el diskette completo. El signo "   " utilizado con SAVE (GUARDAR) como SAVE AND REPLACE (GUARDAR Y REEMPLAZAR), no funcionarán con los archivos relativos.

FORMATO PARA CREAR UN ARCHIVO RELATIVO:

OPEN archivo # , dispositivo # , canal # , "O:nombre,L, " + CHR$(rl #) (longitud del registro)

EJEMPLOS:

OPEN 2,8,2,"O:FILE,L" + CHR$(100) (la longitud del registro es 100)

OPEN F,8,F,"O:" + A$ + ",L," + CHR$(Q)

FORMATO PARA ABRIR ARCHIVOS RELATIVOS EXISTENTES:

OPEN archivo # , dispositivo # , canal # , "O: nombre "

EJEMPLO:

OPEN 2,8,6, "O:TEST"

En este caso, el DOS puede saber mediante la sintaxis que es un archivo relativo. Ambos formatos antedichos permiten la lectura o grabación en el archivo.

SIN EMBARGO, a fin de leer o grabar, ANTES DE CUALQUIER OPERACION, usted debe posicionar el indicador de archivo (file pointer) en la posición correcta del registro.

POSITION (POSICION)

PROPOSITO: Posicionar el indicador de archivo en un registro.

FORMATO: PRINT # archivo # ,"P" CHR$(canal #)CHR$(rec # lo) CHR$(rec # hi)CHR$(posición del registro).

NOTA: CHR$ (posición del registro) especifica la ubicación dentro del registro mismo y es opcional.

Debido a que existen 720 registros disponibles y el número más elevado que puede contener un byte es de 256, deben usarse dos bytes para especificar la posición. El rec # lo contiene la parte menos significativa de la dirección (address) y rec # hi contiene la más significativa. La relación está representada por: rec # = rec # hi * 256 + rec # lo. La rec # es la posición real en un registro cuando comienza la transferencia de datos.

EJEMPLOS:

PRINT # 15,"P"CHR$(2)CHR$(1)CHR$(0)

PRINT # 15,"P"CHR$(CH)CHR$(R1) CHR$(R2)CHR$(P)

A continuación se indica un programa modelo para crear un archivo relativo:

```
10 OPEN 15,8,15
20 OPEN 8,8,8,"O:TEST,L," + CHR$(50)
30 PRINT#15,"P"CHR$(8)CHR$(4)
   CHR$(1)
40 PRINT 8,CHR$(255)
50 CLOSE8:CLOSE15
```

Este programa crea un archivo relativo llamado TEST (PRUEBA) que contendrá registros de 50 bytes de longitud. La línea 30 mueve el indicador a la primera posición en el registro # 1024 (rec# = 256 * 4 + 0 # 1024). Observe que el comando POINTER (INDICADOR) se envía en el canal de comando, mientras que los datos se envían en un canal de datos, 8 en este caso. Debido a que el registro no existía, se generará un mensaje de error, advirtiéndole que no utilice GET# O INPUT# .

Una vez que existe un archivo relativo, usted puede ABRIRLO (OPEN) y expandirlo o acceder al mismo para transferencia de datos. El archivo puede expandirse pero no podrá cambiarse la longitud del registro. Para expandir un archivo, simplemente especifique un número mayor de registros, como en la Línea 30 del programa modelo anterior. Para grabar datos en un archivo relativo ya existente, utilice los siguiente:

```
10 OPEN 15,8,15
20 OPEN 2,8,6,"O:TEST"
30 GOSUB 1000
40 IF A=100 THEN STOP
50 PRINT# 15,"P"CHR$(6)CHR$(100)
   CHR$ (0)CHR$(1)
60 GOSUB 1000
70 IF A=50 THEN PRINT# 2,1:GOTO50
80 IF A=100 THEN STOP.
90 PRINT# 2,"123456789"
100 PRINT# 15,"P"CHR$(6)CHR$(100)
    CHR$(0)CHR$(20)
110 PRINT# 2, "JOHN QWERTY"
120 CLOSE 2: CLOSE15
130 END
1000 INPUT# 15,A,A$,B$,C$
1010 IF (A=50) OR (A<20) THEN RETURN
1020 PRINT "FATAL ERROR:";
1030 PRINT A,A$,B$,C$
1040 A=100:RETURN
```

Las líneas 10 y 20 abren el comando y un canal de datos.

Las líneas 30 y 40 verifican la existencia de errores.

La línea 50 mueve el indicador del archivo a la posición 100 del registro.

Debido a que aún no existen registros, se genera una señal de error.

Las líneas 60, 70 y 80 verifican la existencia de error y crean 100 registros.

La línea 90 graba 9 bytes de datos en las primeras 9 ubicaciones en el registro 100.

La línea 110 imprime luego un nombre desde esa posición.

Es importante que los datos se graben en el registro en forma secuencial a fin de que no se destruyan los datos ya existentes en el registro.

El siguiente programa lee los datos ingresados en el archivo mediante el programa antedicho.

```
10 OPEN 15,8,15
20 OPEN 2,8,6,"O:TEST"
30 GOSUB 1000
40 IF A=100 THEN STOP
50 PRINT# 15,"P"CHR$(6)CHR$(100)
   CHR$(0)CHR$(1)
60 GOSUB 1000
70 IF A=50 THEN PRINT A$
80 IF A=100 THEN STOP
90 INPUT# 2,D$: PRINT D$
100 PRINT#15,"P"CHR$(6)CHR$(100)
    CHR$(0)CHR$(20)
110 INPUT#2,E$: PRINT E$
120 CLOSE 2:CLOSE15
130 END
1000 INPUT# 15,A,A$,B$,C$
1010 IF (A=50) OR (A<20) THEN RETURN.
1020 PRINT "FATAL ERROR:";
1030 PRINT A,A$,B$,C$
1040 A=100:RETURN
```

Las líneas 90, 100 y 110 leen el registro y muestran su contenido en la pantalla. Observe que el retorno del carro enviado al diskette después de cada instrucción PRINT en la rutina de grabación es el separador para cada campo en el registro.

Si el archivo debe ser grabado o leído secuencialmente, no es necesario ajustar el indicador para cada registro. El indicador de registro comienza automáticamente en la Posición 1 si no se ha definido ninguna otra posición. El indicador se mueve a través del registro a medida que cada campo se lee o graba.

9) PROGRAMACION DEL CONTRALOR DE DISKETTE

El FSD-1 es un periférico inteligente, lo que significa que contiene su propio microprocesador y memoria. Un programador avanzado puede tener acceso al microprocesador y a su memoria, suministrando así una amplia gama de aplicaciones. Se pueden diseñar tareas que residen en la memoria del diskette y operan en el microprocesador a fin de controlar la operación de la unidad de discos. Se pueden agregar programas DOS provenientes del diskette propiamente dicho.

La unidad de discos contiene 16K de ROM así como 2K de RAM. El área más útil del programador avanzado es la de la memoria intermedia RAM (buffer RAM) situada entre 4000H y 5FFFH (la H significa que se trata de un número hexadecimal). Se puede grabar en esta área mediante instrucciones de Lenguaje de Máquina y ejecutarse mediante el controlador de diskette (microprocesador).

El método para el manejo de las transferencias de datos a y desde la memoria se denomina comandos de MEMORIA (MEMORY). Existen tres comandos básicos de MEMORY y otros adicionales denominados USUARIO (USER).

MEMORY-WRITE (GRABACION-MEMORIA)

PROPOSITO: Transfiere hasta 34 bytes de datos a la memoria de la unidad.

FORMATO: PRINT #15,"M-W:"CHR$ (dirección byte bajo) CHR$(dirección byte alto)CHR$ (# de caracteres) CHR$ (datos)

MEMORY-WRITE (GRABACION-MEMORIA) le permite grabar hasta 34 bytes de datos por vez en la memoria del controlador de diskette. MEMORY-EXECUTE (EJECUCION-MEMORIA) y USER (USUARIO) pueden usarse para correr este código. Los bytes bajos y altos son el equivalente decimal de la dirección hexadecimal en el espacio de memoria propiamente dicho. El número de bytes corresponde a la cantidad decimal de bytes a ser transferidos, hasta 34. Los datos deben ser la representación decimal de la instrucción codificada en hexadecimal que usted desea enviar. Vea el ejemplo que se indica a continuación.

```
10 OPEN 15,8,15
20 PRINT# 15,"M-W:"CHR$(0)CHR$(112)
   CHR$(3)CHR$(169)CHR$(8)CHR$(96)
30 CLOSE 15
```

Esta rutina graba tres bytes en las posiciones 7000H, 7001H y 7002H (256*112 + 0 = 28672 = 7000H). Los tres bytes son:

169 (A9H, una instrucción de PAGINA CERO (PAGE ZERO)

8 (8H, una posición)

96 (60H, una instrucción de RETORNO DEL CARRO (RETURN).

Cuando se ejecuta, este programa puede hacer que el controlador de la unidad cargue su acumulador con el contenido de la posición 0008H y luego retorne el control nuevamente a la unidad de discos.

MEMORY-READ (LECTURA-MEMORIA)

PROPOSITO: Leer datos de la memoria de la unidad.
FORMATO: PRINT # 15 archivo #, "M-R:" CHR$ (dirección byte bajo) CHR$ (dirección byte alto)

El comando MEMORY READ (LECTURA-MEMORIA) selecciona un byte que debe ser leído desde una posición en la memoria de la unidad de discos, especificada por los bytes altos y bajos de la dirección de posición. La próxima lectura de byte (usando GET = #) del canal # 15, se efectuará desde la posición especificada de memoria. El siguiente ejemplo ilustra lo antedicho mediante la lectura de datos de 10 bytes consecutivos, situados desde FFOOH hasta FFOAH (en decimal, 65280 a 65290).

```
10 OPEN 15,8,15
20 FOR A = 1 TO 10
30 PRINT# 15, "M-R:"CHR$(A)CHR$(255)
40 GET# 15,A$:PRINT ASC(A$ +
   CHR$(0));
50 NEXT
60 CLOSE 15
```

Al utilizar MEMORY-READ (LECTURA-MEMORIA), cualquier uso de INPUT = (ENTRADA #) en el canal de error dará resultados extraños. Esto puede borrarse usando cualquier otro comando, a excepción de los comandos MEMORY. A continuación se indica un programa útil que lee la memoria del controlador de diskette:

```
10 OPEN 15,8,15
20 INPUT"LOCATION PLEASE";A
30 A1 = INT(A/256) : A2 = A - A1*256
40 PRINT# 15, "M-R."CHR$(A2)CHR$(A1)
50 FOR L = 1 TO 5
60 GET# 15,A$
70 PRINT ASC(A$ + CHR$(0))
80 NEXT
90 INPUT"CONTINUE";A$
100 IF LEFT$(A$,1)="Y" THEN 50
110 GOTO 20
```

MEMORY-EXECUTE (EJECUCION-MEMORIA)

PROPOSITO: Ejecuta el programa en la memoria del diskette.
FORMATO: PRINT # 15 archivo #,"M-E:"CHR$(dirección byte bajo) CHR$(dirección byte alto)

Una vez que un programa se ha cargado en la memoria del diskette (ya sea 16K en la ROM ó 2K en la RAM), la dirección del comando MEMORY-EXECUTE (EJECUCION-MEMORIA) especifica dónde comenzará la ejecución del programa. El uso de este comando requiere que el programa a ser ejecutado finalice con un instrucción RTS, a fin de que el control retorne al DOS. A continuación se indica una rutina que escribe un RTS (Return from Subroutine = Retorno de Subrutina).

```
10 OPEN 15,8,15, "M-W:"CHR$(0)CHR$(5)
   ;1;CHR$(96)
20 PRINT # 15,"M-E:"CHR$(0)CHR$(19):
   REM JUMPS TO BYTES, RETURNS
30 CLOSE 15
```

COMANDOS USSER (USUARIO)

Además de los comandos USER1 y USER2 detallados en el capítulo 7, hay otros que, cuando se ejecutan, originan saltos a posiciones específicas en la memoria intermedia (buffer) de la unidad de discos. Esto le permite efectuar rutinas que operan en la memoria del diskette junto con una tabla de saltos, aún en BASIC.

COMANDO USER	FUNCION
U1 ó UA	BLOCK-READ (LECTURA-BLOQUE) sin cambiar el indicador de memoria intermedia (buffer-pointer)
U2 ó UB..................	.. BLOCK-WRITE (GRABACION-BLOQUE) sin cambiar el indicador de memoria intermedia (buffer-pointer)
U3 ó UC	salto a 0500H
U4 ó UD..................	.. salto a 0503H
U5 ó UE	salto a 0506H
U6 ó UF..................	.. salto a 0509H
U7 ó UG	salto a 050CH
U8 ó UH..................	.. salto a 050FH
U9 ó UI	salto a FFFAH
U; ó UJ..................	.. vector de encendido
UI +	ajuste de la velocidad de la Commodore 64
U-	.. ajuste de velocidad VIC 20

EJEMPLOS DE COMANDOS USER (USUARIO)

```
PRINT # 15,"U3"
PRINT # 15,"U" + CHR$(50 + Q)
PRINT # 15,"UI"
```

10-1] CAMBIO DEL NUMERO DE DISPOSITIVO

Todos los periféricos necesitan números de dispositivos para que la computadora pueda identificar a o desde cuál de ellos usted desea transferir datos. El OC-118 está preajustado dentro de su hardware con el número de dispositivo 8, número de unidad 0. El diskette reconoce su propio número de dispositivo observando un puente del hardware en la plaqueta de circuitos y grabando el número basado en el puente, en una sección de su RAM.

El número de dispositivo puede cambiarse mediante dos métodos, hardware y software. Si usted emplea temporariamente dos unidades de discos, el uso del método del software le permite cambiar el número de dispositivo de una unidad temporariamente. Si usted debe utilizar dos (o más) unidades en forma permanente, el método del hardware es una forma simple y permanente para cambiar el número de dispositivo de la unidad.

METODO DEL SOFTWARE

El número de dispositivo se cambia efectuando un MEMORY-WRITE (GRABACION-MEMORIA) en las posiciones 0077H y 0078H. El comando se ejecuta una vez que el canal de comando se haya abierto.

```
FORMATO: PRINT# archivo # ,"M-W:"
         CHR$(119)CHR$(0)CHR$(2)
         CHR$(dirección + 32)CHR$
         (dirección + 64)
```

La dirección es el nuevo número de dispositivo deseado. A continuación se indica un ejemplo para cambiar el número de dispositivo a 9.

```
10 OPEN 15,8,15
20 PRINT# 15,"M-W:"CHR$(119)CHR$(0)
   CHR$(2) CHR$(9+32)CHR$(9+64)
30 CLOSE 15.
```

En primer lugar, encienda una unidad y cambie su número de dispositivo, luego haga lo mismo con la otra unidad, hasta que todas las unidades estén encendidas.

10-2] CAMBIO DEL NUMERO DE DISPOSITIVO

Todos los periféricos necesitan números de dispositivos para que la computadora pueda identificar a o desde cuál de ellos usted desea transferir datos. El número de dispositivo del FSD-1 es muy fácil de cambiar. No se requieren modificaciones en el hardware o software. Simplemente ajuste los interruptores DIP situados en la parte inferior de la unidad para cambiar el número de dispositivo de la unidad. A continuación se indican instrucciones más detalladas.

PARA CAMBIAR EL NUMERO DE DISPOSITIVO:

1. APAGUE LA UNIDAD DE DISCOS.
2. DELO VUELTA Y LOCALICE DOS PEQUEÑOS INTERRUPTORES DIP EN EL MEDIO HACIA LA PARTE TRASERA.
3. AJUSTE LOS INTERRUPTORES CON LA COMBINACION DEL NUMERO DE DISPOSITIVO DESEADO.
   NUMERO DE DISPOSITIVO SELECCIONADO POR:

NUMERO DE DISPOSITIVO:	8	9	10	11
INTERRUPTOR 1:	ON	OFF	ON	OFF
INTERRUPTOR 2:	ON	ON	OFF	OFF

4. LA UNIDAD DE DISCO ESTA AHORA LISTA PARA USARSE CON EL NUEVO NUMERO DE DISPOSITIVO.

APENDICE A. LISTA DE COMANDOS

CAPITULO 4. USO DE PROGRAMAS
- LOAD (CARGA)
- SAVE (GUARDAR)
- SAVE AND REPLACE (GUARDAR Y REEMPLAZAR)
- VERIFY (VERIFICAR)

CAPITULO 5. COMANDOS DEL DISKETTE
- OPEN (ABRIR)
- PRINT # (IMPRIMIR #)
- INITIALIZE (INICIALIZAR)
- NEW (NUEVO)
- SCRATCH (SUPRIMIR)
- COPY (COPIAR)
- RENAME (CAMBIAR EL NOMBRE)
- VALIDATE (VALIDAR)
- CLOSE (CERRAR)

CAPITULO 6. ARCHIVOS SECUENCIALES
OPEN (ABRIR)
PRINT# (IMPRIMIR#)
GET# (OBTENER#)
INPUT# (ENTRADA#)

CAPITULO 7. ARCHIVOS ALEATORIOS
OPEN (ABRIR)
BLOCK-READ (LECTURA-BLOQUE)
BLOCK-WRITE
(GRABACION-BLOQUE)
BLOCK-ALLOCATE
(ASIGNACION-BLOQUE)
BLOCK-FREE
(LIBERACION-BLOQUE)
BUFFER-POINTER (INDICADOR-MEMORIA INTERMEDIA)
USER1 (USUARIO1)
USER2 (USUARIO2)

CAPITULO 8. ARCHIVOS RELATIVOS
POSITION (POSICION)

CAPITULO 9. PROGRAMACION DEL CONTROLADOR DE DISKETTE
MEMORY-WRITE
(GRABACION-MEMORIA)
MEMORY-READ
(LECTURA-MEMORIA)
MEMORY-EXECUTE
(EJECUCION-MEMORIA)
COMANDOS DE USUARIO

APENDICE B. DESCRIPCION DE LOS MENSAJES DE ERROR

Cuando se genera una señal de error, la luz LED en el panel delantero del OC-118 comenzará a centellear. La unidad de discos no enviará el mensaje de error a la computadora a menos que se requiera. La siguiente rutina ingresa el mensaje de error y lo muestra en la pantalla de la computadora.

```
10 OPEN 15,8,15
20 INPUT# 15,A,A$,B,C$
30 PRINT A,A$,B$,C$
40 CLOSE 15
50 END
```

A continuación se indica una lista con las respectivas explicaciones de los mensajes de error usados en la unidad de discos Drean Comm 320:

0: NO ERROR (NINGUN ERROR)
Esta no es una indicación de error y aparecerá cuando se lea el canal de error mientras el LED no esté centelleando.

1: FILES SCRATCHED (ARCHIVOS SUPRIMIDOS)
Esta tampoco es una condición de error. La lectura del canal de error después de haber suprimido uno o dos archivos indicará esto, así como el número de archivos que han sido suprimidos.

2-19: UNUSED ERROR MESSAGE NUMBERS (NUMEROS DE MENSAJES DE ERROR NO USADOS)

20: READ ERROR (ERROR DE LECTURA) (cabecera de bloque no hallada).
El controlador de diskette no puede localizar la cabecera del bloque requerido. Esto puede producirse debido a una cabecera defectuosa en el diskette o por la especificación de un número de sector no válido.

21: READ ERROR (ERROR DE LECTURA) (ningún carácter síncrono).
El controlador de diskette no puede detectar una marca síncrona en la pista deseada. Puede ser causado por la desalineación del cabezal de lectura/grabación o porque el disco no está colocado, no está formateado o no está colocado correctamente. También puede indicar una falla del hardware.

22: READ ERROR (ERROR DE LECTURA) (no está presente al bloque de datos).
Se solicitó al controlador del diskette leer o verificar un bloque de datos que no estaba grabado adecuadamente. Este mensaje de error se produce junto con los comandos de BLOCK (BLOQUE) e indica una solicitud no válida de pista y/o sector.

23: READ ERROR (ERROR DE LECTURA) (error de suma de verificación en el bloque de datos).
Este mensaje de error indica que hay un error en uno o más bytes de datos. Los datos se leyeron en la memoria del DOS, pero la suma de verificación sobre los datos es errónea. También puede indicar problemas de puesta a tierra.

24: READ ERROR (ERROR DE LECTURA) (error de decodificación de bytes).
Los datos o la cabecera han sido leídos en la memoria del DOS, pero se ha creado un error del hardware debido a una configuración inválida de bits en el byte de datos. También puede indicar problemas de puesta a tierra.

25: WRITE ERROR (ERROR DE GRABACION) (error de verificación-grabación).
Este mensaje se genera si el controlador detecta una diferencia entre los datos grabados en la memoria del DOS.

26: WRITE PROTECT ON (PROTECCION DE GRABACION CONECTADA).
Se ha solicitado al controlador grabar un bloque de datos mientras que el interruptor de protección está presionado. Generalmente esto ocurre cuando se utiliza un diskette con una lengüeta de protección sobre la ranura.

27: READ ERROR (ERROR DE LECTURA) (error de suma de verificación en la cabecera).
Hay un error en la cabecera del bloque de datos solicitado. El bloque no se leyó en la memoria del DOS. También puede indicar problemas de puesta a tierra.

28: WRITE ERROR (ERROR DE GRABACION) (bloque de datos largo).
El controlador trata de detectar una marca síncrona de la próxima cabecera después de grabar un bloque de datos. Si la

marca síncrona no aparece dentro de un período predeterminado, se produce un mensaje de error. El error puede originarse por un mal formato del diskette (los datos se extienden hacia el próximo bloque) o por una falla del hardware.

29: DISK ID MISMATCH (DIFERENCIA DE ID DEL DISKETTE).
Se solicitó al controlador acceder a un diskette que no ha sido inicializado o que tiene una cabecera errónea. También ocurre si los diskettes se cambian durante la transferencia de datos.

30 SYNTAX ERROR (ERROR DE SINTAXIS) (sintaxis general).
El DOS no puede interpretar el comando enviado al canal de comando. Generalmente esto se produce por un número no válido de nombres de archivos o porque las configuraciones se utilizan en forma equivocada.

31: SYNTAX ERROR (ERROR DE SINTAXIS) (comando no válido).
El DOS no reconoce el comando. El comando debe comenzar en la primera posición.

32: SYNTAX ERROR (ERROR DE SINTAXIS) (línea larga).
El comando ingresado es superior a 58 caracteres.

33: SYNTAX ERROR (ERROR DE SINTAXIS) (nombre de archivo no válido).
La comparación de configuraciones se utiliza equivocadamente en el comando OPEN (ABRIR) o SAVE (GUARDAR).

34: SYNTAX ERROR (ERROR DE SINTAXIS) (no se indica archivo).
Se omitió el nombre del archivo del comando o el DOS no lo reconoce como tal. Generalmente se omite los dos puntos (:).

35-38: NOT USED (NO USADO).

39: SYNTAX ERROR (ERROR DE SINTAXIS) (comando no válido).
Puede producirse si el comando enviado al canal de comandos no es reconocido por el DOS.

40-49: NOT USED (NO USADO).

50: RECORD NOT PRESENT (REGISTRO NO PRESENTE).
Resultado de la lectura del diskette por el último registro mediante los comandos INPUT o GET. Este mensaje también puede producirse después de posicionarse en un registro más allá del final de un archivo en un archivo relativo. Si se intenta ampliar el archivo agregando el nuevo registro (con un comando PRINT), el mensaje de error puede ignorarse. INPUT o GET no deben usarse después de producido este error sin efectuar primero el reposicionamiento.

51: OVERFLOW IN RECORD (SOBRECARGA EN REGISTRO).
La instrucción PRINT excede los límites del registro, truncando información. Debido a que el retorno del carro (ingresado como fin del registro) se cuenta como parte del tamaño del registro, este mensaje aparecerá si el total de caracteres en el registro (incluyendo el retorno del carro

final) excede el tamaño definido.

52: FILE TOO LARGE (ARCHIVO DEMASIADO GRANDE).
La posición del registro dentro de un archivo relativo indica que se producirá una sobrecarga del diskette.

53-59: NOT USED (NO USADOS).

60: WRITE FILE NOT OPEN (ARCHIVO DE GRABACION NO ABIERTO).
Un archivo de grabación que no ha sido cerrado se abre para lectura.

61: FILE NOT OPEN (ARCHIVO NO ABIERTO).
Un archivo al que se está accediendo no ha sido abierto en el DOS. A veces, en esta situación no se genera un error y la solicitud simplemente se ignora.

62: FILE NOT FOUND (ARCHIVO NO ENCONTRADO).
El archivo solicitado no existe en la unidad indicado.

63: FILE EXISTS (ARCHIVO EXISTE).
El nombre de archivo del archivo que se está creando ya existe en el diskette.

64: FILE TYPE MISMATCH (DIFERENCIA DEL TIPO DE ARCHIVO).
El tipo de archivo no corresponde al tipo de archivo ingresado en el directorio.

65: NO BLOCK (NINGUN BLOQUE).
Ocurre cuando un bloque que debe ser asignado ya lo ha sido. Los parámetros indican la pista y sector disponibles con el próximo número más alto. Si los parámetros corresponden a cero, todos los bloques con un número mayor están en uso.

66: ILLEGAL TRACK AND SECTOR.
El DOS ha intentado acceder a una pista o sector que no existe en el formato que se está usando. Puede indicar un problema de lectura del indicador al próximo bloque.

67: ILLEGAL SYSTEM T OR S (SISTEMA T o S NO VALIDO).
Este error especial indica una pista o sector ilegales del sistema.

68-69: NOT USED (NO USADOS).

70: NO CHANNEL (NINGUN CANAL) (disponible).
El canal solicitado no está disponible o todos los canales están en uso. Un máximo de cinco archivos secuenciales pueden abrirse de una vez en el DOS. Los canales de acceso directo pueden tener seis archivos abiertos.

71: DIRECTORY ERROR (ERROR DE DIRECTORIO).
El BAM (Block Availability Map = Mapa de disponibilidad de bloques) no corresponde al conteo interno. Existe un problema en la asignación del BAM o se excedió la grabación del BAM en la memoria del DOS. Para corregir este problema, reinicialice el diskette para restituir el BAM en la memoria. Algunos archivos activos pueden finalizarse con esta acción correctiva.

72: DISK FULL (DISKETTE LLENO).
Los bloques en el diskette se han usado todos o el directorio ha llegado al límite de las 144 entradas.

73: DOS MISMATCH (DIFERENCIA DEL DOS).

DOS 1 y 2 son compatibles en la lectura pero no en la grabación. Los diskettes pueden leerse indistintamente con cualquier DOS, pero el diskette formateado en una versión no puede grabarse con la otra versión porque el formato es diferente. El error se indica cuando se intenta grabar en un diskette que ha sido formateado con un formato no compatible. Este mensaje puede aparecer también después del encendido.

74: DRIVE NOT READY (DRIVE NO LISTO).

Se ha intentado acceder a la unidad de disco cuando no hay ningún diskette en él.


EL MATERIAL EXHIBIDO
ES SOLO PARA USO
EDUCATIVO, NO COMERCIAL


ENCICLOPEDIA DE
REFERENCIA | EDUCATIVA
RETRO TECNIA
PARA ENTENDER LA EVOLUCION DE LOS ORDENADORES A TRAVES DEL TIEMPO